AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listing of claims in the application.

Listing of Claims:

1. (currently amended) A system for addressing a vector of elements in a memory having a plurality of partitions, the system comprising:

a first storage element for storing a STRIDE value denoting the separation between elements of the vector of elements within each partition of the plurality of partitions;

a second storage element for storing a SKIP value related to the separation between the last element of the vector of elements in one partition of the plurality of partitions and the first element of the vector of elements in the next partition of the plurality of partitions;

a third storage element for storing a SPAN value denoting the number of elements the vector of elements within each partition of the plurality of partitions; and

an arithmetic unit coupled to the first, second and third storage elements and operable to calculate the address in the memory of a next element of the vector of elements ~~dependent upon the values stored in the first, second and third storage elements and the address~~

~~of a current element~~ from the address of a current element, the calculation comprising adding a multiple of the SKIP value to the address of the current element if the next element is in a different partition to the current element and adding a multiple of the STRIDE value to the address of the current element if the next element is in the same partition as the current element.

2. (original) A system in accordance with claim 1, wherein the STRIDE and SKIP values denote a number of elements and further comprising:

a fourth storage element for storing a TYPE value denoting the size of each element of the vector of elements;

wherein the arithmetic unit is coupled to the fourth storage element and is operable to calculate the address in the memory of a next element of the vector of elements dependent upon the values stored in the first, second, third and fourth storage elements and the address of a current element.

3. (original) A system in accordance with claim 1, wherein operation of the arithmetic unit is dependent upon an initial element address EA_START that is indicative of the address of the first element in the vector of elements.

4. (original) A system in accordance with claim 3, wherein operation of the arithmetic unit is further dependent upon the number of elements LEFT_START in the first partition of the plurality of partitions.

5. (original) A system in accordance with claim 2, wherein the system is operable to address a specified number of elements.

6. (original) A system in accordance with claim 2, wherein the arithmetic unit is operable to receive a control instruction indicative of the TYPE, STRIDE, SKIP and SPAN values.

7. (original) A system in accordance with claim 1, wherein the arithmetic unit is operable to receive a control instruction indicative of the address of the first element EA_START, the number of elements LEFT_START in the first partition of the plurality of partitions and the total number of elements TOTAL to be addressed.

8. (original) A system in accordance with claim 1, wherein the arithmetic unit includes:

a first counter for counting a LEFT value indicative of the number of elements remaining in a current memory partition;

a second counter for counting a COUNT value indicative of the total number of elements still to be accessed.

9. (original) A system in accordance with claim 8, wherein the arithmetic unit is operable to reset the first counter when the end of a partition is reached.

10. (currently amended) A system ~~in accordance with claim 1~~ for addressing a vector of elements in a memory having a plurality of partitions, the system comprising:

a first storage element for storing a STRIDE value denoting the separation between elements of the vector of elements within each partition of the plurality of partitions;

a second storage element for storing a SKIP value related to the separation between the last element of the vector of elements in one partition of the plurality of partitions and the first element of the vector of elements in the next partition of the plurality of partitions;

a third storage element for storing a SPAN value denoting the number of elements the vector of elements within each partition of the plurality of partitions; and

an arithmetic unit coupled to the first, second and third storage elements and operable to calculate the address in the memory of a next element of the vector of elements dependent upon the values stored in the first, second and third storage elements and the address of a current element,

wherein the plurality of partitions includes first level partitions and second

level partitions and the arithmetic unit includes:

a fourth storage element for storing a SKIP2 value related to the separation between the last element of the vector of elements in a first level partition and the first element of the vector of elements in a second level partition;

a fifth storage element for storing a SPAN2 value denoting the number of vectors in each second level partition;

a first counter for counting a LEFT value indicative of the number of elements remaining in a current first level memory partition;

a second counter for counting a LEFT2 value indicative of the number of elements remaining in a current second level memory partition; and

a third counter for counting a COUNT value indicative of the total number of elements still to be accessed.

11. (currently amended) A processing system operable to access a partitioned memory, the processing system comprising:

a processing unit having a plurality of functional elements;

an external interface;

an input unit coupled to the processing unit and the external interface and operable to retrieve a vector of elements from the memory via the external interface and pass them to the processing unit, the input unit having a set of input storage elements for storing STRIDE, SKIP and SPAN values and an input arithmetic unit operable to calculate the address in the memory of a next element of the vector of elements dependent upon the STRIDE, SKIP and SPAN values and the address of a current element;

an output unit coupled to the processing unit and the external interface and operable to retrieve a result value from the processing unit and pass it to the external interface; and

a program sequencer coupled to and operable to control the processing unit, the input unit and the output unit.

wherein the STRIDE value denotes the separation between elements of the vector of elements within each partition of the partitioned memory, the SPAN value denotes the number of elements the vector of elements within each partition of the plurality of partitions, and the SKIP value denotes the separation between the last element of the vector of elements in one partition of the partitioned memory and the first element of the vector of elements in the next partition of the partitioned memory and

wherein the input arithmetic unit is operable to calculate the address the next element of the vector of elements from the address of a current element by adding a multiple of the SKIP value to the address of the current element if the next element is in a different partition to the current element and adding a multiple of the STRIDE value to the address of the current element if the next element is in the same partition as the current element.

12. (currently amended) A processing system in accordance with claim 11, wherein the input unit has an additional input storage element for storing a TYPE value and wherein the input arithmetic unit is operable to calculate the address in the memory of a next element of the vector of elements ~~dependent upon the TYPE, STRIDE, SKIP and SPAN values and the address of a current element~~ from the address of a current element by adding the product of the TYPE value with the SKIP value to the address of the current element if the next element is in a different partition to the current element and adding the product of the TYPE value with the STRIDE value to the address of the current element if the next element is in the same partition as the current element.

13. (original) A processing system in accordance with claim 11, wherein the output unit is operable to store a vector of elements received from the processing unit to the memory via the external interface, the output unit having a set of output storage elements to store TYPE, STRIDE, SKIP and SPAN values and an output arithmetic unit operable to calculate the address in the memory of a next element of the vector of elements dependent upon

the TYPE, STRIDE, SKIP and SPAN values and the address of a current element.

14. (original) A processing system in accordance with claim 11, wherein the external interface is a memory interface and further comprising a memory unit coupled to the memory interface.

15. (original) A method for accessing a vector of elements in a memory having a plurality of partitions, comprising:

accessing the memory at an element address in a partition;

stepping a first counter; and

if a second counter indicates that at least one vector element remains in the partition:

incrementing the element address by a first amount; and

stepping the second counter;

otherwise:

incrementing the element address by a second amount; and

resetting the second counter to indicate the number of elements of the vector of elements in a partition.

16. (original) A method in accordance with claim 15, wherein the first amount is a product of the size of an element and the number of memory elements between adjacent vector elements in a partition.

17. (original) A method in accordance with claim 15, wherein the first amount is a difference between consecutive vector element addresses in a partition.

18. (original) A method in accordance with claim 15, wherein the second amount is a product of the size of an element and the number of elements between last element of one partition and the first element of the next partition.

19. (original) A method in accordance with claim 15, wherein the second amount is a difference between the last vector element address in one partition and the first vector element address in the next partition.

20. (original) A method in accordance with claim 15, wherein the element address is initialized as the address of the first element in a partition and wherein the first counter is initialized to indicate the number of elements of the vector of elements in the first partition of the plurality of partitions, thereby allowing memory access to begin part way through the first partition.

21. (original) A method in accordance with claim 15, wherein accessing the memory at the element address in a partition comprises reading a value of

appropriate size stored at the element address.

22. (original) A method in accordance with claim 15, wherein accessing the memory at the element address in a partition comprises storing a value of appropriate size at the element address.

23. (original) A method in accordance with claim 15, wherein the first counter, the second counter and the element address are not re-initialized if the memory access is interrupted and resumed before all of the elements of the vector have been accessed.

24. (original) A method in accordance with claim 15, wherein the first amount and the second amount are set to default values unless otherwise specified.

25. (original) A method for accessing a vector of elements in a memory having a plurality of first and second level partitions, comprising:

accessing the memory at an element address in a first level and second level partition of the plurality of first and second level partitions;

stepping a first counter;

incrementing the element address by a first amount; and

if a second counter indicates that at least one vector element remains

in the first level partition:

stepping the second counter;

otherwise:

incrementing the element address by a second amount;

resetting the second counter to indicate the number of elements of the vector of elements in the next first level partition; and

if a third counter indicates that at least one vector element remains in the second level partition:

stepping the third counter;

otherwise:

incrementing the element address by a third amount;

resetting the third counter to indicate the number of elements of the vector of elements in the next second level partition.